Task-6:- Procedures, Functions and Loops in PL/SQL (Based on Online Food ordering system)

Case study: Online Food ordering system.

Objective: The objective of this task is to design implement, and execute PL/SQL procedures, function, and loops to handle real-world business scenarios related to an online food ordering system. This will helps in automating transactions, improving database efficiency, and enforcing business rules in a structed manner.

Step 1: Ensure the Necessary Tables Exist

Before running the procedures and function, create the required tables in your oracle Database.

```
DROP TABLE orderTable PURGE;
DROP TABLE Delivey PURGE;
DROP TABLE Menu_Item PURGE;

CREATE TABLE orderTable (
    Order_ID NUMBER PRIMARY KEY,
    Cust_ID NUMBER,
    Order_Date DATE,
    Order_Total NUMBER(10,2),
    payment_status VARCHAR(20);
);
CREATE TABLE Delivery (
    order_ID NUMBER PRIMARY KEY,
    Delivery_status VARCHAR(20)
    FOREIGN KEY (order_ID)
    REFERENCES orderTable(order_ID)
);
```

~~Query 2 : Exec~~

Query 2: Function to Calculate Total Revenue

Step 1 : Create a Function.

```
CREATE OR REPLACE FUNCTION Get_Total_
    Revenue RETURN NUMBER AS
    V_Total_Revenue NUMBER;
BEGIN
    SELECT SUM(order_Total) INTO V_Total_Revenue
      FROM order Table;
    RETURN V_Total_Revenue;
END;
/
```

Expected Output

Function created.

Step 2 : Execution.

```
GET_TOTAL_REVENUE()
801.25
```

```
CREATE TABLE Menu-Item (
    Item-ID NUMBER PRIMARY KEY,
    Item-Name VARCHAR2(100)
    Price Number(10,2)
);
INSERT INTO OrderTable Values (1,101, To-DATE
    ('2024-02-101', 'YYYY-MM-DD'), 250.50,
    'Pending');
INSERT ITO OrderTable VALUES (2,102, To DATE
    ('2024-02-12', 'YYYY-MM-DD'), 400.75, 'Paid');
INSERT INTO OrderTable VALUES (3,103, To-DATE (
    ('2024-02-03', 'YYYY-MM-DD'), 150.80, 'Pending');

INSERT INTO Delivery VALUES (1, 'Pending');
INSERT INTO Delivery VALUES (2, 'Delivered');
INSERT INTO Delivery VALUES (3, 'Pending');
```

Query 3. Loop: Mark All undelivered Orders as "Delayed".

```
DECLARE
    v_order_ID OrderTable.Order_ID%TYPE;
    CURSOR Cur IS SELECT order_ID FROM
      Delivery WHERE Delivery_status = 'pending';
BEGIN
    OPEN Cur;
    Loop
      FETCH Cur INTO v_order_ID;
      EXIT WHEN cur%NOTFOUND
      UPDATE Delivery
       SET Delivery_Status = 'Delayed'
       WHER Order_ID = v_order_ID;
       DBMS_OUTPUT.PUT_LINE('Order '|| v_order_ID||
         ' marked as Delayed.');
    END Loop;
    CLOSE cur;
    COMMIT;
END;
/
```

Expected Output:

1 row(s) updated.

Query 5: Procedure to Apply Discount on Menu Items

Step 1: Create a produre:

```
CREATE OR REPLACE PROCEDURE Apply_Discount(
    discount_percent IN NUMBER
)
IS
BEGIN
    UPDATE Menu_Item
    SET Price=price-(Price*discount_percent/100);
    COMMIT;
    DBMS_OUTPUT.PUT_LINE('Discount Apply:'||
        discount_percent||'%');
END;
/
```

Expected Output:

Procedure created.

Step 2: Execution:

```
BEGIN
    Apply_Discount(10);
END;
/
```

Expecoded Output:

Discount Applied: 10%

Statement processed

VELTECH	
EX No.	6
PERFORMANCE (5)	5
RESULT AND ANALYSIS (3)	3
VIVA VOCE (3)	3
RECORD (4)	4
TOTAL (15)	15
SIGN WITH DATE	8/9/25

Result:

Hence, Implementing Procedures, functions and loops in PL/SQL has been done successfully.

8/9/25